

Bases des données avancées

Contenu du Cours

- Généralités
- Environnement Oracle
 - Installation Serveur Oracle
 - Installation et configuration client Oracle
 - Interface SQL*PLUS
- Structures logiques de stockage
 - Tablespaces, Segments et Extensions
 - Gestion de tablespaces
- Programmation procédurale
 - Principes Fondamentaux
 - Fonctions & Procédures Stockées
 - Déclencheurs
 - Curseurs
- Gestion des utilisateurs Oracle

Généralités

Les compétences visées dans ce chapitre consistent à clarifier les concepts clés des bases des données, présenter l'architecture applicative du système de gestion des bases des données Oracle et faire connaître les principes transactionnels

Ce chapitre vise à clarifier :

- Notions de base
- Architecture logicielle
- Principes de transaction

Notions de base

Qu'est-ce qu'une Base des données ?

Collecte de données structurée, bien organisée et gérée au moyen d'un logiciel appelé Système de gestion des bases des données.

une base des données peut être définie de manière:

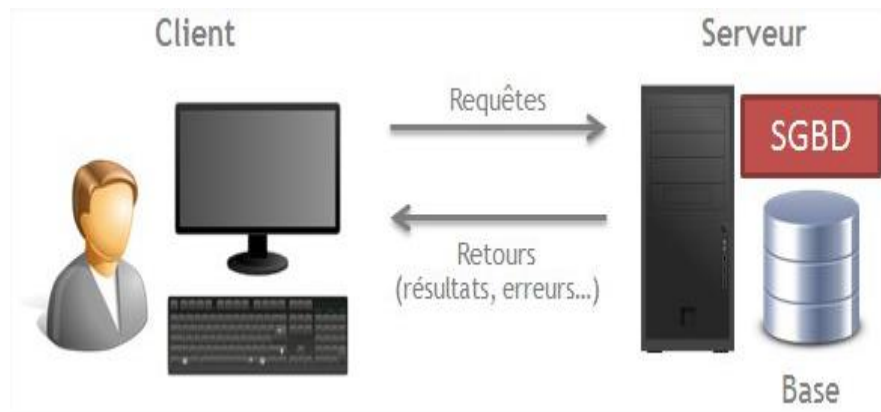
Logique: il sert d'ensemble d'objets logiques, gérée au moyen du langage SQL,

Physique: ensemble de fichiers physiques sur le disque, accessible seulement par des méthodes d'accès propres au SGBD.

Architecture Applicative

La plupart du temps, les SGBD fonctionnent sur le modèle « client-serveur » (vous pouvez en voir une illustration sur la figure ci-dessous). On distingue alors les deux composantes suivantes :

- le client, qui envoie ses requêtes au serveur ;
- le serveur, dont la tâche est d'attendre les requêtes du client, avant de les traiter et de lui envoyer une réponse



Serveur Oracle:

Il a pour rôle de :

- Gérer et contrôler les bases de données
- Traiter les requêtes SQL
- Renvoyer le résultat d'exécution, etc.

Client Oracle:

Il a pour rôle de :

- Demander l'établissement de connexion à la BD
- Soumettre les requêtes SQL au serveur
- Présenter le résultat d'exécution des requêtes SQL

Principes de transaction

Transaction est un bloc des opérations indissociable, totalement accompli ou annulé,

Depuis certain temps très reculé, le principe de transaction est implémenté dans les quasi-totalités de SGBD,

En général, une fois l'opération exécutée avec succès, elle passe automatiquement en attente de validation, puis la confirmation ou l'annulation sera explicitement appliquée.

Environnement Oracle

Ce chapitre consiste à fournir les compétences techniques nécessaires pour installer de différents produits Oracle et effectuer les configurations requises pour la mise en place et l'exploitation d'une base des données oracle.

Installation du Serveur Oracle

Le processus d'installation comprend généralement trois phases:

Pré-installation: Préparer le système d'exploitation

Installation: Installer les produits oracle

Post-installation: Terminer l'installation et configurer certains composants oracle

a. Pré-installation

consiste à effectuer des tâches préliminaires qui nécessitent une bonne compréhension de l'architecture interne et des compétences minimales sur le système d'exploitation,

Ces compétences sont réduites au strict minimum pour la plate-forme **windows**, mais sont un peu plus avancées pour les autres plate-formes,

Sur la plate-forme **windows**, la phase pré-installation est réduite aux :

- ✓ vérification de prérequis logiciels et matériels
- ✓ accès par le compte d'un membre du groupe Administrateur

Sur la plate-forme **linux**, cette phase comporte, par contre, plusieurs étapes.

Dans les grandes lignes, les étapes sont les suivantes:

- ✓ vérifier les prérequis logiciels et matériels;
- ✓ préparer le noyau;
- ✓ créer les répertoires nécessaires;
- ✓ créer un groupe et un compte appartenant à ce groupe

b. Installation

L'installation des produits oracle s'effectue avec l'application **Oracle univesal installer**, il permet :

- de choisir le type d'installation: Entreprise Edition, Standard Edition, personnel Edition(uniquement pour windows)
- de créer une base de données de départ avec différentes options de configuration

À l'issue de cette phase, et une fois l'installation est opté pour base des données de départ, la base des données et le processus d'écoute (listener) seront lancés automatiquement,

- sur plate-forme windows, OUI crée automatiquement les services et les configure en démarrage et arrêt automatique,

- Sur plate-forme linux, les services doivent explicitement être créés et configurés par l'administrateur du système d'exploitation

Oracle Universel Installer, en mode interactif, doit être lancé sur un environnement **X WINDOX**,

Si l'affichage X ne s'effectue sur la machine sur laquelle est installé oracle, la variable DISPLAY doit être positionnée pour déporter l'affichage sur une autre machine.

Le processus d'installation du serveur de bases des données est décrit en détail dans un guide des installations préparé en dehors de ce support de cours.

c. Post-installation

Elle consiste essentiellement à:

- télécharger et appliquer d'éventuels patches Oracle ;
- configurer certains composants Oracle (Oracle Net, etc.) ;
- installer des produits supplémentaires ;
- configurer l'environnement de travail ;
- configurer le démarrage et l'arrêt automatiques des différents composants Oracle (base de données, processus d'écoute, etc.)

.

Installation et configuration du client

Le client oracle est une application, installée localement sur le poste client, permettant l'accès distant à une ou plusieurs bases des données déployées auprès d'un serveur de bases données oracle,

a. Installation

Il peut s'installer sur la quasi-totalité des systèmes d'exploitation, mains pour des raisons simples liées à la mise en place, on se limite à la plateforme windows,

Le processus d'installation du client Oracle est décrit en détail dans le même guide des installations déjà utilisé pour le serveur Oracle

b. Configuration

À la fin de l'installation du client Oracle, les actions suivantes seront requises :

- Définition des variables d'environnement **ORACLEHOME**, **TNS_ADMIN**, etc.
- Mise en place du fichier tnsname.ora

Le fichier tnsname.ora doit contenir ce type de déclaration:

PROTECT =

(DESCRIPTION =

(ADDRESS_LIST =

(ADDRESS = (PROTOCOL = TCP)(HOST = <ip_address>)(PORT = <port_number>))

)

(CONNECT_DATA =

(SERVICE_NAME = protect)

)

)

Interface SQL*PLUS

L'interface SQL*PLUS sert de moyen de communication entre les utilisateurs et le serveur, notamment les bases des données oracle,

Elle permet, entre autres, de :

- établir la connexions à la base des données
- écrire et exécuter des requêtes SQL et blocs PL/SQL
- présenter le résultat d'exécution des requêtes SQL et blocs PL/SQL

SQL*PLUS est disponible en deux modes(lignes de commandes et graphique),mais à partir de la version de la version 11g n'est disponible que le monde en ligne de commandes. le répertoire **BIN** contient tous les programmes fournis par oracle, y compris **sqlplus**, qui se charge de lancer l'interface **SQL*PLUS** en mode console,

Au cas où le **PATH** a été correctement défini, il suffit tout simplement de lancer la commande **sqlplus [/nolog]** pour démarrer l'interface **SQL*PLUS** avec ou non la demande d'établissement de connexion à une base des données Oracle

a. Connexion à la base des données Oracle

L'utilisateur, avant d'aller chercher à se connecter à la base des données Oracle, doit premièrement savoir quel type de connexion (locale, distante) à utiliser,

Ces deux situations se présentent respectivement dans le cas où le client et le serveur sont déployés sur la même machine ou sont physiquement séparés.

Mode locale

Ce mode porte uniquement sur la manipulation de la variable ORACLE_SID qui a pour but de définir le nom de la base des données oracle que les utilisateurs peuvent utiliser, donc il suffit d'affecter à la variable ORACLE_SID la valeur correspondante, puis finalement lancer sur la console SQL*PLUS la commande suivante: **CONN[ECT] login/password**

Mode distant

Ce mode est appliqué réellement si la base de données cible est hébergée sur un serveur distant, ce qui signifie que l'adresse réseau de la base de données Oracle doit être renseignée lors de l'établissement de la connexion.

b. Commandes Spécifiques

Oracle comme pratiquement tout les SGBD a aussi leur commandes Spécifiques pour:

- La connexion à la base des données
- Gestion des variables
- Gestion du buffer
- Paramétrage de l'affichage
- ...,

b. Commandes Spécifiques

Oracle comme pratiquement tout les SGBD a aussi leur commandes
Spécifiques pour:

CONN DEF ACCEPT START
EDIT LIST RUN / CHANGE
SAVE COLUMN SPOOL, etc ...,

c. Paramètres d'affichage

PAGESIZE LINESIZE TIMING TIME
SQLPROMPT

Structures logiques de Stockage

Ce chapitre consiste à fournir au DBA des compétences très poussées en administration, lui permettant de bien gérer et contrôler finement l'espace disque alloué aux différents objets de la base des données afin d'optimiser le stockage physique ainsi que d'augmenter la performance du serveur Oracle lui même

I. Tablespaces, Segments et extensions

a. Tablespaces

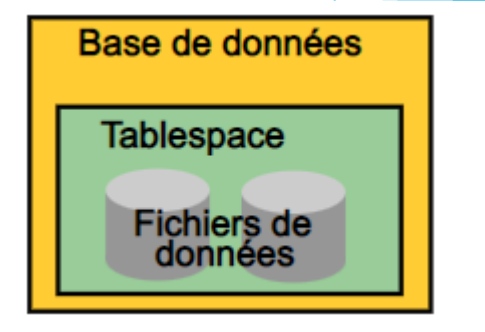
un tablespace Oracle est structure logique servant à décrire la manière dont sont stockées physiquement les données d'une base des données Oracle,

un tablespace comprend un ou plusieurs fichiers de données

En fonction du contenu de tablespaces, il en existe deux types:

- Tablespace SYSTEM: contient le dictionnaire de données
- Tablespace NON SYSTEM: contient les données utilisateurs

À l'intérieur d'un tablespace, les données sont organisées en segments contenant plusieurs extensions(extents), une extension étant un ensemble de blocs contigus



b. Segments

un segment sert de structure des données destinée à stocker les données d'un objet, comme table, d'une base des données,
une fois l'objet créé, Oracle lui associe automatiquement un segment

Lorsqu'un segment est créé dans un tablespace, Oracle lui alloue une ou plusieurs extensions dans un des fichiers de données du tablespace.

Lorsque l'espace initialement alloué est plein (suite à l'insertion de données par exemple), Oracle alloue une nouvelle extension au segment, et ainsi de suite.

Toutes les extensions allouées à un segment sont dans le tablespace de création du segment, mais pas obligatoirement côte à côte, ni forcément dans le même fichier de données (si le tablespace est composé de plusieurs fichiers de données),

Pour rappel, il existe quatre types principaux de segments:

- **Segment de table** : espace occupé par les tables ;
- **Segment d'index** : espace occupé par les index ;
- **Segment d'annulation** : espace temporaire utilisé pour stocker les information permettant d'annuler une transaction ;
- **Segment temporaire** : espace temporaire utilisé notamment lors d'un tri.

Tablespaces, Segments et Extensions

c. Extensions et blocs

Extensions

une extension est un ensemble de blocs Oracle contigus et présent au sein du même fichier appartenant au tablespace,

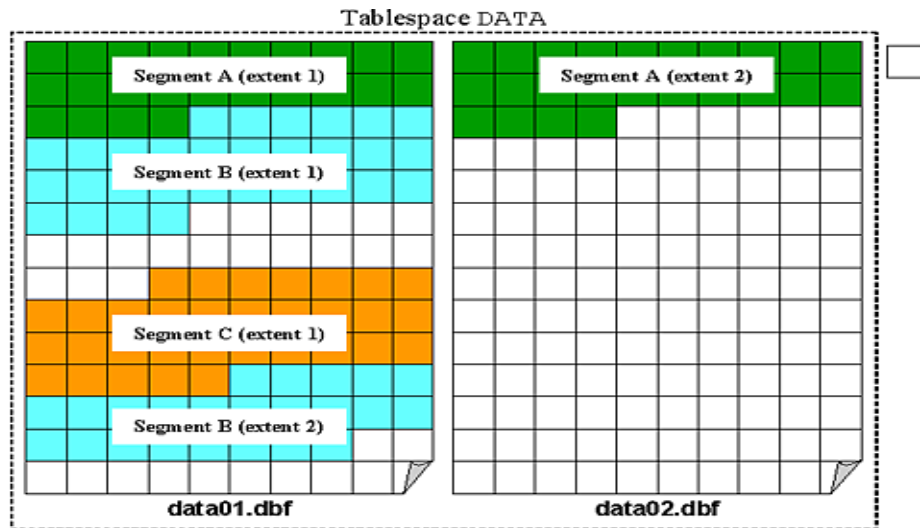
La première extension d'un segment contient au minimum deux blocs, le premier étant réservé à l'en-tête du segment (ne contient pas de données utiles mais la carte des extensions allouées au segment).

Blocs Oracle

un bloc est un ensemble de lignes d'un objet oracle, dont la taille est un multiple de la taille d'un bloc du système d'exploitation,

Structures logiques de stockage
Tablespaces, Segments et Extensions

En conclusion, les données sont organisées dans le tablespace comme illustré ci-dessous:



Bloc Oracle
Blanc= libre
Autre couleur= alloué

II. Gestion de tablespaces

Cette partie de cours porte sur :

- La création de tablespaces
- La Gestion de tablespaces

Oracle stocke logiquement les données dans les tablespaces et physiquement dans des fichiers des données,

a. Création des tablespaces

```
CREATE TABLESPACE userdata
```

```
DATAFILE 'chemin_file' SIZE nombre {K|M|G}
```

```
[AUTOEXTEND ON NEXT nombre {K|M|G} MAXSIZE nombre {K|M|G} ];
```

b. Tablespaces d'annulation

- Il permet de stocker des segments d'annulation
- Il ne peut contenir aucun autre objet

Un tablespace d'annulation est crée de la sorte suivante:

```
CREATE UNDO TABLESPACE undo1
```

```
DATAFILE ' chemin' SIZE nombre taille;
```

c. Tablespaces TEMPORARY

- Ils sont utilisés pour les opérations de tri
- Ils ne peuvent contenir d'objets permanents
- La gestion locale des extents est recommandée

Tablespaces TEMPORARY

CREATE TEMPORARY TABLESPACE tbtmp

TEMPFILE 'chemin' SIZE taille

EXTENT MANAGEMENT LOCAL UNIFORM SIZE taille

d. Tablespaces TEMPORARY par défaut

Après la création de la base des données

ALTER DATABASE DEFAULT TEMPORARY temps;

Les tablespaces TEMPORARY ne peuvent pas être:

- Supprimés tant qu'un nouveau tablespace par défaut n'est pas disponible
- Mise hors ligne

Tablespaces accessibles en lecture seule

On utilise la commande suivante pour mettre hors ligne un tablespace

ALTER TABLESPACE tbsnom READ ONLY;

Mise hors ligne d'un tablespace

ALTER TABLESPACE tbsnom OFFLINE;

Mise en ligne d'un tablespace

ALTER TABLESPACE tbsnom ONLINE;

Modification des paramètres de stockage

**ALTER TABLESPACE tbsnom
DEFAULT STORAGE(
INITIAL taille NEXT taille
MAXEXTENTS nombre);**

e. Redimensionnement d'un tablespace

un tablespace peut être redimensionné:

- automatiquement à l'aide de AUTOEXTEND
- Manuellement à l'aide de ALTER TABLESPACE

f. Déplacement d'un fichier de données

un fichier de données d'un tablespace peut être déplacé à l'aide

ALTER TABLESPACE ou ALTER DATABASE :

ALTER TABLESPACE tbs_name

RENAME DATAFILE cheminancien TO cheminrecent;

ALTER DATABASE RENAME FILE cheminancien TO cheminrecent;

g. Suppression d'un tablespace

un tablespace ne peut pas être supprimé:

- S'il s'agit du tablespace SYSTEM
- S'il possède un segment actif

INCLUDING CONTENTS: supprime les segments

INCLUDING CONTENTS AND DATAFILES supprime les fichiers de données

CASCADE CONSTRAINTS supprime les contraintes d'intégrités référentielles

DROP TABLESPACE tbsnom **INCLUDING CONTENTS AND DATAFILES;**

Programmation Procédurale

Les compétences visées dans ce chapitre sont généralement la capacité à combiner, dans les mêmes lignes de code, les fonctions apportées par les langages procéduraux (structures conditionnelles, récurrentes, etc.) et celles du SQL, sans recourir à des langages de programmation externes.

I. Principes Fondamentaux

a. Définitions

PL/SQL est un langage procédurale, dérivé du SQL, qui permet de réaliser des traitement très complexes sur les bases de données. les fonctions offertes sont les mêmes qu'avec des langages impératifs.

un programme PL/SQL se termine obligatoirement par /, sinon le SQL*Plus ne l'interprète pas. S'il contient des erreurs de compilation, il est possible d'afficher les messages d'erreur avec la commande SQL*Plus : **SHOW ERRORS.**

a. Blocs

Tout bloc PL/SQL comprend éventuellement une section de déclaration de variables et obligatoirement une section de traitement dans la quelle les variables déclarées sont visibles. La syntaxe:

```
DECLARE      } Section de déclaration
BEGIN      }
END;         } Section de traitement
/
```

b. Affichage

Pour afficher les valeurs de variables, il faut faire appel aux procédures:

DBMS_OUTPUT.PUT_LINE() et **DBMS_OUTPUT.PUT()**

c. Variables

La variable doit se déclarer comme suit:

nom_variable TYPE [:=valeur];

l'initialisation est optionnelle et les types peuvent être aussi les mêmes qu'en SQL

d. Traitement conditionnel

IF et CASE fonctionnent de la même façon que dans les autres langages impératifs

```
IF /*condition1*/ THEN  
/*instructions1*/  
ELSE /*instructions2*/  
END IF ;
```

```
IF /*condition1*/ THEN  
/*instructions1*/  
ELSIF /*condition2*/  
/*instructions2*/  
ELSE /*instructions3*/ 37  
END IF ;
```

e. Traitement répétitifs

LOOP END LOOP permet d'implémenter les boucles, et l'instruction **EXIT WHEN** est requise pour en sortir

```
LOOP /* i n s t r u c t i o n s */  
EXIT WHEN /* c o n d i t i o n */ ;  
END LOOP ;
```

La boucle **FOR** existe aussi en PL/SQL

```
FOR /* v a r i a b l e */ IN  
/* i n f */ .. /* s u p */ LOOP  
/* i n s t r u c t i o n s */  
END LOOP ;
```

Ainsi que la boucle **WHILE** :

```
WHILE /* c o n d i t i o n */ LOOP  
/* i n s t r u c t i o n s */  
END LOOP ;
```

f. Structures

une structure est un type regroupant plusieurs types, une variable de type structuré contient plusieurs variables, ces variables s'appellent aussi champs, on définit un type structuré de la sorte:

```
TYPE /* nomType */ IS RECORD  
  
(  
  /* l i s t e de s champs */  
);
```

Il est possible de se référer au type d'une colonne de table avec le type:
nomTable.nomColonne%type

g. SELECT INTO

Ce type de requête a introduit une interaction entre la base des données et le script PL/SQL,

```
SELECT lst_champs INTO lst_var  
FROM nom_table ;
```

h. Exceptions

Le Mécanisme des exceptions est implémenté dans la plus part des Langages récents, notamment orientés objet,

La syntaxe est de la sorte de:

EXCEPTION

WHEN E1 THEN /* t r a i t e m e n t */

WHEN E2 THEN /* t r a i t e m e n t */

WHEN E3 THEN /* t r a i t e m e n t */

WHEN OTHERS THEN /* t r a i t e m e n t */

END;

a. Fonctions

La syntaxe de création d'une fonction est :

```
CREATE OR REPLACE FUNCTION /* nom */ ( /* p a r a m e t r e s */ )
```

```
RETURN /* t y p e */ IS
```

```
/* déclaration des variables */
```

```
BEGIN
```

```
/* i n s t r u c t i o n s */
```

```
RETURN valeur ;
```

```
END;
```

III. Déclencheurs

a. Définition

un déclencheur, appelé aussi TRIGGER, est une procédure stockée qui se lance automatiquement lorsqu'un événement se produit.

Par évènement, on entend dans ce cours toute modification des données se trouvant dans les tables. On s'en sert pour contrôler ou appliquer des contraintes qu'il est impossible de formuler de façon déclarative

b. Classification

Types d'évènements

Les évènements, dans ce cours, portent sur les :

INSERT UPDATE DELETE

Moment de l'exécution

AFTER | BEFORE

On précise aussi si le trigger doit être exécutée avant (**BEFORE**) ou après (**AFTER**) l'évènement.

Evènements non atomiques

par exemple, Lorsqu'on fait un **DELETE**, il n'y a qu'une seule instruction, mais plusieurs lignes sont affectées, il existe donc deux types :

- **FOR EACH ROW TRIGGER:** est exécuté à chaque fois qu'une ligne est affectée
- **STATEMENT TRIGGER:** est exécutée à chaque fois qu'une instruction est lancée.

Syntaxe de création d'un trigger

On déclare un trigger avec l'instruction suivante :

```
CREATE OR REPLACE TRIGGER nomtrigger [ BEFORE | AFTER ] [INSERT |  
DELETE | UPDATE] ON nomtable [ FOR EACH ROW | ]  
DECLARE  
/* declarations */  
BEGIN  
/* instructions */  
END;
```

c. Accès aux lignes en cours de modification

Dans les **FOR EACH ROW** triggers, il est possible avant la modification de chaque ligne, de lire l'ancienne ligne et la nouvelle ligne par l'intermédiaire des deux variables structurées **:OLD** et **:NEW**.

IV. Curseurs

a. Définitions

un curseur est un objet mémoire permettant de stocker le résultat d'une requête de sélection quelque soit sa taille,

b. Déclaration

un curseur se déclare dans une section **DECLARE**:

```
CURSOR /* nomcurseur */ IS /* requête */ ;
```

c. Ouverture

Lors de l'ouverture d'un curseur, la requête du curseur est évaluée, et le curseur contient toutes les données retournées par la requête. On ouvre un curseur dans une section **BEGIN** :

```
OPEN /* nomcurseur */ ;
```

Par exemple,

DECLARE

CURSOR emp_cur IS SELECT * FROM EMP ;

BEGIN

OPEN emp_cur ;

/* utilisation du curseur*/

END;

Lecture d'une ligne

Une fois ouvert, le curseur contient toutes les lignes du résultat de la requête.

On les récupère une par une en utilisant le mot-clé

FETCH :

FETCH /* nom curseur */ INTO

/* liste de variables */ ;

Si la lecture de la ligne échoue, parce qu'il n'y a plus de ligne à lire, l'attribut **%NOTFOUND** prend la valeur vrai.

Syntaxe:

DECLARE

CURSOR emp_cur IS SELECT * FROM EMP ;

ligne emp_cur%rowtype

BEGIN

OPEN emp_cur ;

LOOP

FETCH emp_cur INTO ligne ;

EXIT WHEN emp_cur%NOTFOUND ;

BMS_OUTPUT . PUT_LINE (ligne . ename)

END LOOP ;

CLOSE /* nomcursus */ ;

END;

Avec la boucle **WHILE** :

OPEN emp_cur ;

FETCH emp_cur INTO ;

WHILE emp_cur%FOUND **LOOP**

FETCH emp_cur INTO.....;

END LOOP ;

Avec la boucle **FOR** :

FOR ligne IN emp_cur **LOOP**

/* Traitement*/

END LOOP ;

Gestion des utilisateurs Oracle

Connexion

L'accès à la base des données Oracle est lié à la possession d'un compte d'utilisateur ayant les privilèges nécessaires.

Les actions qui peuvent être exécutées par un utilisateur donné dépendent des privilèges qui lui ont été accordés

- **Création d'un utilisateur Oracle**

La syntaxe de création d'un compte d'utilisateur Oracle

```
CREATE USER login IDENTIFIED BY password  
DEFAULT TABLESPACE name_tbs  
TEMPORARY TABLESPACE name_tmp  
QUOTA nombre {K|M|G|UNLIMITED} ON name_tbs  
PASSWORD EXPIRE  
ACCOUNT {unlock | lock} ;
```

- **Assignment de privilèges**

La syntaxe d'attribution d'un ou plusieurs droits

```
GRANT {privilège | list_privilèges} TO name_user;  
list_privilèges= privilège, privilège[, ... ,privilège]
```

- **Assignment de rôle**

```
GRANT rôle To name_user;
```

- **Privilège WITH ADMIN OPTION**

Si un utilisateurs reçoit explicitement avec un privilège système l'option
WITH ADMIN OPTION, donc il peut, à son tour, donner ce privilège même à
d'autres utilisateurs

- **Retrait de privilèges**

REVOKE privilège [,...,privilège] FROM name_user;

- **Gestion de mot de passe**

ALTER USER name_user IDENTIFIED BY password [PASSWORD EXPIRE];

- **Verrouillage/déverrouillage de compte oracle**

ALTER USER name_user ACCOUNT { lock,unlock};

- **Suppression d'un compte oracle**

DROP USER name_user [CASCADE];

■ Privilèges

Les privilèges sont classés en deux types:

- Privilèges Système: ce type de privilège permet d'autoriser l'utilisateur d'effectuer un certain nombre d'opération tel que (se connecter ou bien de supprimer un objet de la base de données) ces privilèges ne sont pas spécifique à un objet de schéma.
- Privilèges objets: ce type de privilège permet d'autoriser l'utilisateur d'effectuer des modification ou à accéder à certains objets de la base de données comme une table ou une vue, etc.

- Liste de certains privilèges système

Catégorie	Exemples
INDEX	CREATE ANY INDEX ALTER ANY INDEX DROP ANY INDEX
TABLE	CREATE TABLE CREATE ANY TABLE ALTER ANY TABLE DROP ANY TABLE SELECT ANY TABLE UPDATE ANY TABLE DELETE ANY TABLE
SESSION	CREATE SESSION ALTER SESSION RESTRICTED SESSION
TABLESPACE	CREATE TABLESPACE ALTER TABLESPACE DROP TABLESPACE UNLIMITED TABLESPACE

- Liste de certains privilèges objets

Privilège Objet	Table	Vue	Séquence	Procédure
ALTER	✓		✓	
DELETE	✓	✓		
EXECUTE				✓
INDEX	✓			
INSERT	✓	✓		
REFERENCES	✓			
SELECT	✓	✓	✓	
UPDATE	✓	✓		